Divergent Token Metrics: Measuring degradation to prune away LLM components – and optimize quantization

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have reshaped natural language processing with their impressive capabilities. Their ever-increasing size, however, have raised concerns about their effective deployment and the need for LLM compression. This study introduces the *Divergent Token* Metrics (DTMs), a novel approach for assessing compressed LLMs, addressing the limitations of traditional perplexity or accuracy measures that fail to accurately reflect text generation quality. DTMs focus on token divergence, that allow deeper insights into the subtleties of model compression, in particular when evaluating components' impacts individually. Utilizing the *First Divergent Token* Metric (FDTM) in model sparsification reveals that 25% of all attention components can be pruned beyond 90% on the Llama-2 model family, still keeping SOTA performance. For quantization FDTM suggests that over 80% of parameters can naively be transformed to int8 without special outlier management. These evaluations indicate the necessity of choosing appropriate compressions for parameters individually—and that FDTM can identify those—while standard metrics result in deteriorated outcomes.

1 Introduction

Cutting-edge Large Language Models (LLMs) based on the transformer architecture (Vaswani et al., 2017) have revolutionized Natural Language Processing with their exceptional performance, notably exemplified by the GPT-series (Radford et al., 2018, 2019; Brown et al., 2020; Bubeck et al., 2023) in text generation. However, these models have grown massive, even exceeding half a trillion parameters (Chowdhery et al., 2023). While the large number of parameters aid early training convergence, their practical utility and true necessity remain unclear.

Compression strategies like sparsification and quantization can enhance parameter efficiency. Current metrics, however, either average too coarsely, such as perplexity, or are by design too specific, such as standard NLP benchmarks. Both fail to capture the diverging performance nuances introduced early on by the compression because they ignore the actual discontinous text generation process. This however is the main use of the final model, and so we argue that they are therefore insufficient measures for the performance of the compressed model. This misalignment can lead to unwanted subtle discrepancies in generation, such as grammatical errors or a mismatch in numbers as we will see, even when overall metrics, such as perplexity, appear satisfactory (*cf.* Prop. 3.2, Sec. 4). An example is depicted in Fig. 1.

To meet these challenges, we introduce the family of *Divergent Token* Metrics (DTMs). These metrics are tailored to measure the *model divergence* of LLMs throughout the compression process and in relation to the actual generation procedure. We demonstrate that the *First Divergent Token* Metric (FDTM) and the *Share of Divergent Tokens* Metric (SDTM) offer a more nuanced evaluation compared to perplexity. They also enable individual component evaluation to rank parts of the model best suited for compression, thus enabling meaningful compression while preserving text generation quality. Specifically, sparsification enhanced by FDTM indicates significant differences in component utilization across layers. For the first time, we show that 25% percent of the models' attention components can be pruned beyond 90%, and several even entirely removed, while preserving a single-digit perplexity. Consequently, one can employ a sparse matrix format to accelerate computational efficiency. Likewise, for precision reduction we show that sorting components by FDTM coincidentally correlates to sorting by their induced number of outliers when being naively converted to int8. FDTM identifies the optimal 80% of components that keep overall performance without spe-

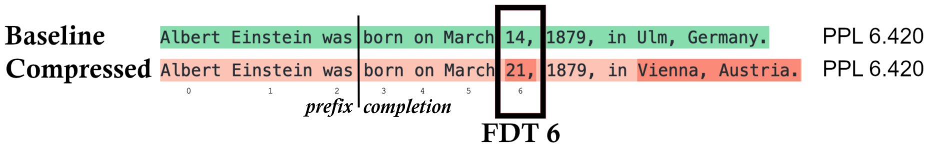


Figure 1: Illustration of a diverging generation process. Given the 3-token prefix as prompt, a baseline and its compressed model generate 8 subsequent tokens. Our proposed metric points to the first divergent token (FDT). The FDT may cause further divergence during the iterative generation process. Note how both models score the same perplexity value, as it does not reflect the actual sampling process (*c.f.* Fig. 2, Sec. 4 for an empirical exploration).

cific outlier-handling. The observed decline in performance with more outliers, and the significant influence of specific components on those, suggests re-evaluating the applied normalization methods throughout the model. We demonstrate that this level of precision goes beyond what standard perplexity and conventional NLP benchmarks can achieve. The proposed Divergent Token Metrics closely reflect the generation process and so can be a measure to foster confidence in the deployed compressed models.

2 Compression Principles

Model compression aims to reduce the hardware resources needed to operate the model. However, doing so may sacrifice model accuracy. To keep the loss as small as possible, a corrective measure is typically used. Here, we discuss the most commonly used concepts and state-of-the-art methods for sparsification and quantization of LLMs.

Outlier and Hessians. Most model compression methods rely either on the separation of outliers (Dettmers et al., 2022; Sun et al., 2023) or the computation of a Hessian matrix (Frantar et al., 2023; Frantar and Alistarh, 2023). Outliers usually refer to significantly larger values in magnitude occurring either in the weight matrix directly or in the activations during a forward pass. As most computations are linear matrix multiplications, such outliers strongly influence the remaining entropy contained in consecutive computations. In the case of sparsification, outliers should be left intact, and the values with the least magnitude—which are consequently the least influential—should be masked instead (Han et al., 2015). On the other hand, Hessian matrices can be applied to correct errors (Frantar et al., 2023). They can effectively be approximated by computing backpropagation gradients for a small number of samples and represent a second-order approximation to reconstruct the original model.

Sparsification. The goal of sparsification is a reduction of the overall number of weights and, hence, a distillation of the relevant computation. Typically, this method is divided into "structured" and "unstructured" pruning. *Structured-pruning* aims to locate dynamics, such as the irrelevance of an entire layer or dimension for a given use case and prunes these entirely. *Unstructured-pruning* usually refers to the masking of weights, i.e., setting the irrelevant weights to 0. High levels of sparse matrix computations could result in more efficient kernels and computations. In scenarios where masks exceed a 90% threshold, the implementation of a specialized sparse matrix format becomes feasible. This format predominantly stores the indices of non-zero weights. While it necessitates some additional storage for these indices, the overall requirement is reduced due to the exclusion of zero values. Moreover, this approach substantially improves computational performance.

Magnitude pruning selects the masking of weights based only on their magnitudes. This is fast to compute but significantly degrades model performance when pruning large amounts simultaneously. To resolve this issue, *wanda* (Sun et al., 2023) proposes to sample a small amount of data and incorporate activations during the forward pass. It was shown that this generates more effective one-shot pruning masks. *SparseGPT* (Frantar and Alistarh, 2023) computes iterative Hessian approximations to select the lowest impact weights and correct the remaining.

Note that the incorporation of activations can to some extent be interpreted as a form of training. Moreover, despite these efforts, one-shot pruning has not yet produced directly usable models without further final fine-tunings. This is in particular the case for the high sparsity levels beyond 70% that we target. Finally, there has not yet been any investigation of the individual components.

Quantization. Model quantization refers to the reduction of the precision of the numeric format used. Usually, LLMs are trained in 16-bit floating-point (fp16) and converted to 8-bit integer (int8) representations. The naive conversion of float matrices to integers is *AbsMax* rounding. This divides a number by the maximum value occurring in the matrix and multiplies by the largest available integer—as such, it spans a uniform representation grid. The largest float value is stored and multiplied for dequantization. The most prominent methods to mitigate the introduced rounding errors are LLM.int8() and GPTQ.

Dettmers et al. (2022) introduced *LLM.int8()*, which identifies vectors containing outliers and retains them in their original fp16 form during the matrix multiplication of a forward pass. The vectors lacking outliers are quantized fully. The int8 weights and activations during the forward pass are subsequently multiplied and dequantized afterward. This allows them to be integrated with the fp16 representation of the outliers. Through empirical investigation optimizing the trade-off between degradation in perplexity and the number of outliers preserved in fp16, they fixed an absolute outlier threshold.

The *GPTQ* framework offers a more robust quantization approach, in particular, to different integer bit-precisions. It does not rely on any outlier detection mechanism or mixed precision computations—matrix multiplications with the weights are fully performed using integers. Frantar et al. (2023) introduce an efficient Hessian approximation and iteratively quantize the weights of the matrices while performing error corrections on the remaining weights.

3 Model Divergence Metrics

Perplexity fails to identify minor variations in model degradation at an early stage. This behavior is depicted in Fig. 1 and 2 and discussed in Sec. 3.5 and 4 in more detail. To assess model divergence and enhance the model compression process, we introduce token-based metrics specifically designed to detect those nuances occurring in early compression stages. We start by establishing our notation and presenting the perplexity metric (PPL). Subsequently, we introduce an enhanced variant of PPL and propose the *Share of Divergent Tokens* Metric (SDTM) and *First Divergent Token* Metric (FDTM). We conclude by discussing the advantages of each metric compared to traditional perplexity-based measures when assessing the degradation of the generative performance of compressed models.

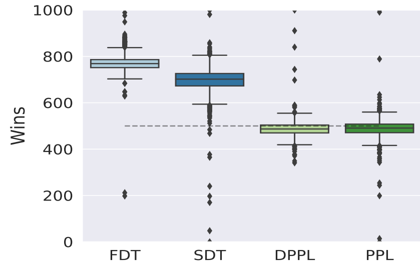


Figure 2: Pruning lowest weights, and random weights. FDT is able to discriminate the cases. PPL exactly performs on the level of guessing. *C.f.* Sec. 4.2.

3.1 Basic notation

Let F denote an auto-regressive model over a vocabulary $\mathcal{V} = \{0, 1, ..., |\mathcal{V}| - 1\}$, $y = (y_1, .., y_N) \in \mathcal{V}^N$ an arbitrary token sequence and $F(y) = F(y)_{ij} \in \mathbb{R}^{N \times |\mathcal{V}|}$ the model logits, with i denoting the sequence and j the respective vocabulary positions. Given a prefix length $n < N$, we denote the token prefix $y_{:n} = (y_1, ..., y_n)$ and the greedily decoded completion up to index N by $\mathcal{G}(F, y_{:n}, N)$. It is defined recursively as follows: $\mathcal{G}(F, y_{:n}, N)_{:n} = y_{:n}$, and for $n \leq i \leq N - 1$

$$\mathcal{G}(F, y_{:n}, N)_{i+1} = \arg\max_j F(\mathcal{G}(F, y_{:n}, N)_{:i})_{ij}.$$

3.2 Perplexity (PPL)

Given a ground truth sequence y and model F, the negative log-likelihood of y given F is

$$\mathrm{NLL}(y, F, n) = -\frac{1}{N-n} \sum_{i=n}^{N-1} \log \mathbb{P}(y_{i+1}|y_i, .., y_1),$$

with $\mathbb{P}(y_{i+1}|y_i, .., y_1) = (\mathrm{softmax}\ F(y))_{iy_{i+1}}$. Then the *perplexity (PPL)* is given by

$$\mathrm{PPL}(y, F, n) = \exp(\mathrm{NLL}(y, F, n)).$$

A common practice in the literature, e.g. (Dettmers et al., 2022), is to measure model degradation as the increase in average perplexity over a given test dataset $\mathcal{D}$, e.g. randomly sampled from C4 (Raffel et al., 2020). Usually, this metric is computed disregarding the prefix, i.e., with $\mathrm{PPL}(y, F) := \mathrm{PPL}(y, F, 1)$.

3.3 Context aware model comparison

First, we argue that standard evaluation does not reflect the typical generative model usage, i.e., there are no empty prompts, and as such, those positions should not be taken into account when evaluating the generative performance. Moreover, when comparing a compressed model F' to the original model F, one is interested to what extent *the original behavior is kept*. Therefore, we propose to use the outputs of the original model F as a ground truth to assess the performance of the compressed model F'. This leads to the definition of the *divergent perplexity (DPPL) metric* as

$$M_{\mathrm{DPPL}}(F,F',y_{:n},N) \qquad (1)$$
$$= \mathrm{PPL}(\mathcal{G}(F,y_{:n},N),F',n) \;.$$

Finally, let $\mathcal{D}$ be an arbitrary test dataset containing documents of potentially varying length. For a fixed prompt length n and completion length N, we define the *aggregated divergent perplexity metric* as the complete evaluation on the dataset:

$$\mathcal{M}_{\mathrm{DPPL}}(F,F',n,N) = \qquad (2)$$
$$\tfrac{1}{|\mathcal{D}|}\sum_{y\in\mathcal{D}} M_{\mathrm{DPPL}}(F,F',y_{:n},N).$$

The DPPL already substantially improves discriminative capabilities over PPL, as we will demonstrate in the empirical evaluation.

3.4 Divergent Token Metrics

SDT. To further improve on the expressiveness and interpretability of model divergence, we propose the *share of divergent tokens (SDT)* as follows:

$$\mathrm{SDT}(y,F,n)$$
$$= |\{i \geq n\colon \; \arg\max_j \; F(y)_{ij} \neq y_{i+1}\}|,$$

$\mathrm{SDT}(y,F,n)$ can be interpreted as the number of times the model would need to be corrected during decoding to match the ground truth after consuming the prefix. This measure provides a more direct interpretation of the errors occurring during actual token generation, as opposed to estimating prediction certainties as PPL does.

FDT. In addition to SDT, we introduce the *first divergent token (FDT)* as

$$\mathrm{FDT}(y,F,n) \qquad (3)$$
$$= \min\{i \geq n\colon \; \arg\max_j \; F(y)_{i,j} \neq y_{i+1}\} - n,$$

with the convention that the minimum is equal to N if the set on the right-hand side is empty. Analogously to Eq. 1 and Eq. 2, we define $M_{\mathrm{SDT}}, M_{\mathrm{FDT}}$, $\mathcal{M}_{\mathrm{SDT}}$ and $\mathcal{M}_{\mathrm{FDT}}$ in the same fashion.

As an illustrative example, consider computing $M_{\mathrm{FDT}}(F,F',y_{:n},N)$. We first perform a greedy decoding of $N-n$ tokens with the base model F given the prefix $y_{:n}$. We then feed the sequence $\mathcal{G}(F,y_{:n},N)$ into the compressed model F' and find the first index greater than or equal to n, where the logit argmax of F' differs from what F generated. This computation can be done in a single forward pass similar to perplexity, and so is more efficient than accuracy based evaluations. Trivially, $0 \leq M_{\mathrm{FDT}}(F,F',y_{:n},N) \leq N-n$, where the upper bound is reached if and only if F and F' would generate the exact same sequence up to position N given the prefix $y_{:n}$. Further note that M_{FDT} is symmetric, i.e. $M_{\mathrm{FDT}}(F,F',y_{:n},N) = M_{\mathrm{FDT}}(F',F,y_{:n},N)$, in contrast to PPL.

In the following, we will ease notation and omit $\mathcal{M}$, or the words aggregated and metric, when they are clear from the context.

3.5 Token vs. Perplexity Metrics

It turns out that divergent token metrics offer a superior criterion for analyzing model performance degradation compared to perplexity-based metrics, especially in the context of greedy decoding. The main reason is that the greedy decoding operation $\mathcal{G}$ is a discontinuous function of the logits. To formalize this, let us discard the model itself and focus notation solely on the concept of logits.

Definition 3.1. *The operators and metrics from previous sections defined for models F, F' are defined for logits $l, l' \in \mathbb{R}^{N\times|\mathcal{V}|}$ by replacing all occurrences of F, F' with l, l'.*

For example, $\mathcal{G}(l,y_{:n},N)_{i+1} = \arg\max_j l_{ij}$, for $n \leq i \leq N$.

Proposition 3.2. *Given any y, N and $\varepsilon > 0$ there exist logits $l, l' \in \mathbb{R}^{N\times|\mathcal{V}|}$ such that*

$$|\,\mathrm{PPL}(y,l,1) - \mathrm{PPL}(y,l',1)| < \varepsilon,$$
$$M_{\mathrm{SDT}}(l,l',y_{:1},N) = N.$$

Proof. See App. A. □

This means that even if the average perplexity of a compressed model matches the perplexity of the original model, the compressed model can produce a very different (and potentially worse) output when performing greedy decoding. Hence leading to a false positive. In practice, this is a severe issue since even a single diverging token can lead to a completely different subsequent output. It is illustrated in Fig. 1 and 2 and discussed in Sec. 4.2.

As described before, another option is to compute the perplexity with respect to the generated completions of the original model. This metric relates more reasonably to the share of divergent tokens (SDT):

Proposition 3.3. *The following upper bound holds:*

$$M_{\mathrm{SDT}}(l, l', y_{:n}, N) \leq \frac{N-n}{\log 2} \log M_{\mathrm{DPPL}}(l, l', y_{:n}, N).$$

Proof. See App. A. □

However, a comparable lower bound does not generally hold. In fact, in the case $l = l'$ we trivially have $M_{\mathrm{SDT}}(l, l, y_{:n}, N) = 0$. Further, the value of $M_{\mathrm{DPPL}}(l, l, y_{:n}, N)$ can still be as high as the maximal value, which occurs when l is a perfectly flat distribution at any sequence index. This could lead to a false negative signal for the generation process.

In conclusion, perplexity-based metrics suffer from false positives or false negatives when evaluating degradation of generative performance. The case for FDT and SDT is quite straightforward in that they both directly measure the difference between model outputs in a *what-you-see-is-what-you-get* manner.

Note that additional token-based metrics, such as the measurement of the distance between erroneous predictions, can be readily formulated. These metrics may prove especially valuable when assessing potential benefits, for instance, in the context of correction-based inference strategies like speculative decoding (Leviathan et al., 2023). We now empirically demonstrate the improvements of well-known compression methods using our metrics.

4 Token Metrics Improve Model Compression

We will demonstrate in the following how the proposed metrics provide novel insights into the efficiency of the architecture of LLMs and establish benchmarks for model compression. Throughout all experiments we outperform standard PPL as a ranking metric.

More precisely, we apply component-wise probing on sparsification to determine individual sparsity rates. Interestingly, the model tends to entirely remove components of the attention mechanism on certain layers. In total 40 out of 160 attention components are sparsed beyond 90% and 15 removed completely. For quantization, on the other hand, we show how component selection significantly influences the overall number of model outliers. For the first time, almost 10% of model components can be converted to 4-bit integer representations without significant model degradation.

4.1 Experimental Protocol

Let us start by clarifying the experimental setup.

Test environment. All experiments were performed on the public Llama2-7B and 13B models (Touvron et al., 2023). Note, however, that we observed similar behavior amongst other decoder transformer models. It remains, in particular, with upscaled sizes, and smaller variations on the architecture or the training procedure.

For all experiments, we follow best practices of compression evaluations (Sun et al., 2023) and randomly sample data from the C4 dataset (Raffel et al., 2020) for training iterations. The final model evaluation is done comparing loss on the Wikitext2 dataset (Merity et al., 2017) and standard NLP benchmarks (Gao et al., 2021).

Metrics. We apply our proposed metrics for performance evaluation as well as selection criteria.

We employ FDT, SDT, DPPL and PPL as metrics to assess the overall model divergence. When it comes to model compression, we demonstrate that both PPL and our variant DPPL typically struggle to measure minor changes adequately (*cf.* Sec. 3.5, 4.2 and Fig. 2). On the other hand, FDT is particularly suited to characterize errors for subtle model changes. Consequently, we apply FDT for model compression. In the following paragraph, we describe the selected parameters for using FDT in more detail.

Divergent Token Parameters. We empirically selected hyperparameters as follows. Through preliminary sparsification experiments, we observed that the most variance is present in the 75%-quantile of FDT, as defined in Eq. 3. We denote this value by FDT_{75}. In the following it is our compare-to value.

Next, we swept over the given *context prefix length* n of FDT and sparsification steps as depicted in Fig. 3 on the y- and x-axis respectively. 'The heatmap shows the overall variance of FDT_{75} on 5k probes. For simplicity, we fixed the prefix length to 100 tokens, as it is most discriminative on average.

We observed that most sparsification steps introduce an error in FDT_{75} within a range of 500 *completion tokens*. Therefore, we selected $N = 500$. Finally, to determine the *number of probes* $|\mathcal{D}|$, we compared the mean deviation against a baseline of 5000 probes. As the deviation of 1000 to 5000

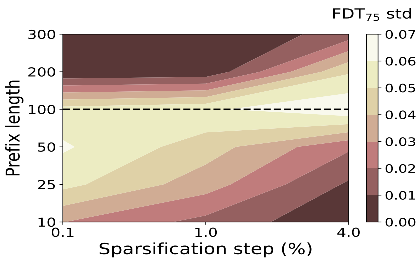


Figure 3: Hyperparameter selection of FDT. Visualized is the std in FDT$_{75}$ over all components when varying prefix length (y-axis) and applying different choices for sparsity-step increases (x-axis), *c.f.* Sec. 4.1 and 4.2.

probes only differs on average by a value 4 in mean FDT$_{75}$, we selected $|\mathcal{D}| = 1000$.

Pruning of LLMs. In Sec. 4.2, we will show that FDT improves sparsification procedures to achieve high compression rates on LLama2-13B. To this end, we iterate small unstructured sparsification with continued training steps for the model to attune to the remaining weights and recover performance. Specifically, we apply eight iterations to increase the average model sparsity by $20, 15, 10, 10, 5, 5, 5$, and 5 percent, resulting in a final model with 25% total parameters remaining.

We run this experiment in two configurations, uniform and FDT-selective. Uniform sparsification applies the target increase of the current round to each component uniformly, pruning the lowest weights. For FDT, we determine individual component sparsification values to evenly distribute the induced error. Based on the previous sparsed model F and for the current target increase $step$, we probe each component c_i separately with an additional $step \pm step/2$ percent of lowest weights pruned, denoted by F^{c_i+s}, to determine its FDT$_{75}$ value. We further add the constant extrema, i.e., step sparsity 0 and 100% with FDT$_{75}$ values of 500 and 0. Given these four data points, we segment-wise interpolate linearly to achieve the highest value of FDT$_{75}$ possible throughout all components, but on average yielding the target sparsity. Specifically, we find the set of component-sparsities $\{s_i\}$ that optimize for

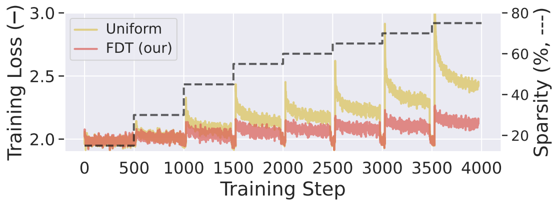
$$\arg\max_{\{s_i\}} \min_i M_{\mathrm{FDT}_{75}}(F, F^{c_i+s_i}),$$

such that $\sum_i \tilde{s}_i = step$ using $\tilde{s}_i$ to represent the normalized sparsity of s_i relative to the individual parameters of component c_i.

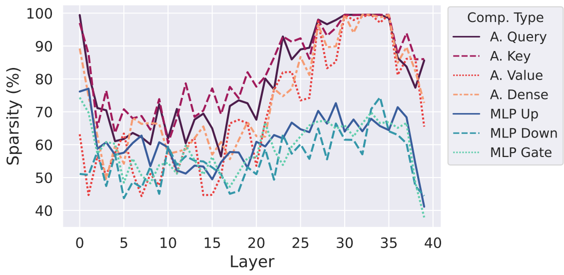
We further follow the findings of AC/DC (Peste et al., 2021) and alternate compressed and decompressed iterations as follows: Each round we train a total of 500 steps, from which the first 450 are with sparsification mask applied, and the following 50 without any masks. We found this alternation to produce smaller spikes in training loss after sparsification steps. This yields a total of 4000 training steps. During training, we apply a weight decay of 0.01, batch size 256, and sequence length 2048.

Note that throughout this experiment series, we only apply pure magnitude pruning per iteration. The probing strategy can be applied to other methods, such as Wanda, as well.

Quantization of LLMs. For model quantization, we compare the performance of the proposed metrics on the task of sorting the model's components by their lowest introduced error. To this end, we build a search tree to find the best model configuration as follows: We construct a parallel depth-first search tree with a branching width of 10. This means that, at each level of the tree, we simultaneously explore all possible successor configs for the currently top-10 performing nodes, with one more component naively quantized using AbsMax. From this newly identified set of nodes, we again select the best-performing 10 nodes for the next

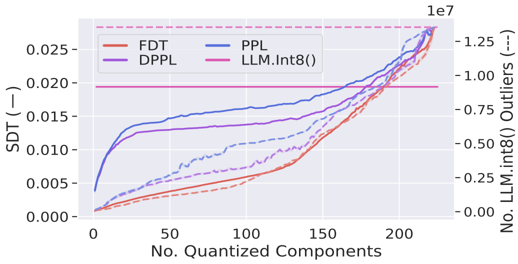


(a) Comparison of uniform and component-wise pruning using FDT as a metric for comparison.

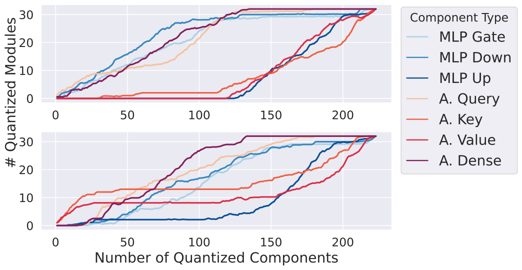


(b) Converged component config with 75% average sparsity. Layers (x-axis), Component-sparsity (y-axis).

Figure 4: Depiction of the proposed sparsification process that converged to a 75% sparse Llama-2-13B. **a)** Model training performance throughout all rounds. Our FDT-based sparsification clearly outperforms uniform magnitude pruning. **b)** Converged sparsity values per component. One quarter of attention components are pruned beyond 90% sparsity. Significant outliers appear in first and last layers.



(a) Performance of FDT vs PPL



(b) Selected Components FDT vs PPL

Figure 5: Evaluation of the Tree Search as described in text. **a)** Comparison of Tree Search based component-wise quantization. Different numbers of components (x-axis) lead to different token divergence scores (y-axis, normalized to $[0, 1]$), and in particular correlates early on to introduced outliers (second y-axis). Throughout the entire search, FDT is able to rank components by their potential errors and, coincidentally, outliers. **b)** Selected components at respective depth. A.Key and A.Value induce most error.

Sparsification

Model	FDT ↑	PPL ↓	NLP ↑
Llama2-13B	-	4.884	53.59
∼ 60% sparse (unif.)	4.7	9.244	46.32
∼ 60% sparse (our)	**7.9**	**6.242**	**48.89**
∼ 75% sparse (unif.)	3.5	13.512	41.67
∼ 75% sparse (our)	**5.5**	**8.101**	**46.32**
∼ 80% sparse (our)	5.2	9.531	45.66

Quantization

	Model	FDT ↑	PPL ↓	NLP ↑
	Llama2-7B	-	5.472	50.79
.int8	LLM.int8()$_{all}$	36.1	5.505	**50.81**
.int8	AbsMax PPL$_{150}$	46.3	5.500	50.72
.int8	AbsMax DPPL$_{150}$	54.1	5.490	50.75
.int8	AbsMax FDT$_{150}$ (our)	**71.7**	**5.489**	50.75
.int4	GPTQ$_{all}$	11.1	5.665	48.34
.int4	GPTQ PPL$_{16}$	45.0	5.511	49.91
.int4	GPTQ DPPL$_{16}$	137.0	5.476	50.02
.int4	GPTQ FDT$_{16}$ (our)	**205.0**	**5.475**	**50.13**

Table 1: Evaluations of Compressed Models. Even when evaluating the final model, standard NLP benchmarks don't reflect the actual model degradation, as observed in AbsMax quantization. FDT, PPL are evaluated on Wikitext2. Subscript refers to best found k quantized components. Bold denote best values.

iteration. Starting with the unquantized base model Llama2-7B, each node contains exactly the number of quantized components respective to its depth, while the final node is a fully AbsMax quantized model. We further apply deduplication to prevent redundant computations.

4.2 Sparsification reveals: Attention is not all you need!

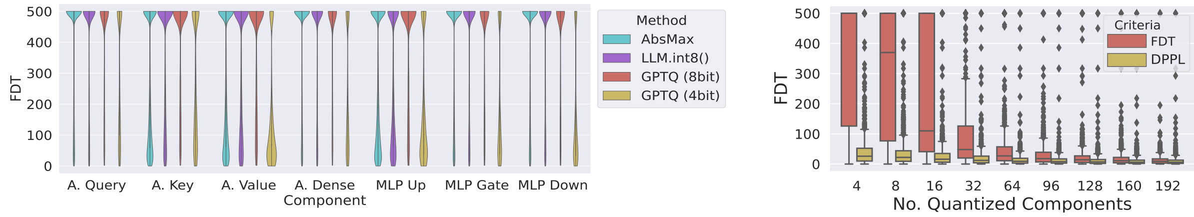
We applied step-wise uniform magnitude pruning, and our balanced component-wise pruning using FDT, to achieve 75% model sparsity. A summary of the results is shown in Fig. 4.

Attention almost erased. Fig. 4b visualizes the converged sparsity values when applying our balanced pruning using FDT. Notably, the model favors pruning attention over MLP. In total 40 out of 160 attention components are sparsed beyond 90% and 15 even completely removed. In general the second half of the model appears to be more prunable than the first half. The value matrices are overall least pruned of the attention matrices. Finally, significant outliers appear at the first and last layers. This finding indicates that attention is not efficiently utilized throughout the entire model. In fact, only layers 3 to 20 and layer 40 appear to be of significant relevance for the model's final prediction. This observation might be attributed to an evolving shift in distributions, and with that the concepts processed in embeddings.

Notably, in the first layer Attention Value and MLP Down remain significantly dense, while all others are comparably sparse. This observation indicates an incomplete shift of token-embeddings.

General Observations. As shown in Fig. 4a, FDT based balanced pruning significantly lowers the introduced error between sparsification rounds. Uniform pruning, on the other hand, substantially diverged, and in particular does not regain performance with the given amount of compute. Generally speaking, what is lost can hardly be recovered.

The standard evaluation of FDT and PPL on Wikitext2, is found in Tab. 1. The 75% compressed 13B model, with several components pruned away, scored PPL 8.1, compared to PPL 4.8 of the base model. Note that no other model sparsed beyond 70% has yet been reported in particular achieving single-digit PPL. Uniform pruning achieved 13.5. Further note, that we almost doubled the mean FDT value when compared to uniform pruning. However, as the generally low FDT value suggests, it still diverged from the base model.



(a) Quantization methods evaluated on Components. (b) Selected GPTQ(4bit) components.

Figure 6: Evaluation of FDT performance. **a)** evaluates components separately on all quantization methods. Clear outliers in performance are A.Value and MLP.up. GPTQ(8bit) is able to evenly amortize the induced error. **b)** Selecting top-k components of GPTQ(4bit). FDT is suited to rank components one-shot.

FDT is more discriminative. In practice, FDT is better able to discriminate subtle changes than than PPL. We demonstrate this with a test as follows: On each component of the model, we prune 0.1% of the weights either randomly or from the lowest weights. The resulting model is probed for 1000 trials with all discussed metrics used to distinguish the cases. The results in Fig. 2 clearly indicate that FDT is able to distinguish the cases, while they remain indifferent for PPL-based comparison. We therefore omit using PPL as a metric to determine step-sizes for the described sparsification experiment.

4.3 Quantization: Outliers can be prevented

Finally, we demonstrate the impact of selecting the right components for quantization. We compare the proposed metrics PPL, DPPL, and FDT as ranking criteria to showcase their discrimation capabilities.

Quantization without outlier-handling. Fig. 5 shows the average performance of the top 10 nodes occuring in the respective search tree depth (x-axis). FDT constantly outperforms the other metrics on the Share of Divergent Token Metric (y-axis). Notably, this is on par with the total number of outliers occurring for the respective configs (second y-axis). Certain components appear to significantly influence the decline observed in both measures. While DPPL enhances some aspects of performance, neither variant of PPL effectively distinguishes these components and tends to select those prematurely.

With FDT, we can cast 80%, *i.e.* 150, of the model's components directly to int8 using only naive AbsMax—and without further outlier handling—still outperforming full LLM.int8() conversion in model performance. Selecting those 150 components with DPPL and FDT leads to close perplexity scores 5.490 and 5.489 on Wikitext2, *c.f.* Tab. 1. However the resulting mean FDT improves by almost 50% when also selecting the components by this metric. The larger generation of the same sequences suggests a model closer to the original when choosing FDT as a selection criterion. Fig. 5b) shows the selected components to each depth respective of a). Most outliers occur when selecting Attention Key early on. Notably, we observed in Sec. 4.2, that this is one of the matrices most suitable to sparsify.

16 components in 4-bit. Figure 6a) presents a comprehensive assessment of the quantization techniques discussed. First, it is noticeable that the LLM.int8() method slightly lowers the lower quantile scores of FDT in comparison to AbsMax. Yet, GPTQ-8bit demonstrates superior performance, outshining both plain AbsMax and LLM.int8(). This method achieves a more balanced error distribution across all components (*c.f.* App. Fig. 17). Conversely, GPTQ-4bit shows noticeable deviations in the generation process, with only a limited number of components achieving FDT scores above 300. Despite this, the discriminative power of FDT enabled us to identify and merge the top 16 components that minimally compromised the model's integrity, as illustrated in Fig. 6b).

5 Conclusion

We introduced the Divergent Token Metrics (DTMs), a tailored approach to evaluate the performance differences of compressed generative models. In particular, DTMs respect the usually applied greedy sampling procedure to generate predictions. We proved that DTMs achieve appropriate metric bounds and are not affected from catastrophic artefacts that perplexity-based metrics encounter. Importantly, using DTMs, we achieved an outperforming 75% sparse version of the Llama2-13B model and successfully converted 80% of the LLama2-7B components naively to int8.

Limitations

With the proposed DTMs, compression processes can be tailored to use cases—and we can measure their performance degeneration. We hinted with the sparsification experiments, that MLP and Attention can be ascribed varying levels of significance throughout the layers. These variations should be further exploited to optimize model architectures. In particular, variations of specific datasets to probe or finetune on could lead to interesting variations.

As a pruning strategy, we achieved outperforming results using only naive magnitude pruning. DTMs should be directly applicable to other masking strategies, such as Wanda (Sun et al., 2023), which may further improve results. Finally, the generalizability of the metrics to other sampling strategies should be investigated.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Túlio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. *CoRR*, abs/2303.12712.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24:240:1–240:113.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm.int8(): 8-bit matrix multiplication for transformers at scale. *CoRR*, abs/2208.07339.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, volume 202, pages 10323–10337.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. OPTQ: accurate quantization for generative pre-trained transformers. In *International Conference on Learning Representations*.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2021. A framework for few-shot language model evaluation.

Song Han, Jeff Pool, John Tran, and William J. Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, pages 1135–1143.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, volume 202, pages 19274–19286.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *International Conference on Learning Representations*.

Alexandra Peste, Eugenia Iofinova, Adrian Vladu, and Dan Alistarh. 2021. AC/DC: alternating compressed/decompressed training of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 8557–8570.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. *OpenAI blog*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:140:1–140:67.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. 2023. A simple and effective pruning approach for large language models. *CoRR*, abs/2306.11695.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Appendix

A Proof of Propositions

Proof of Proposition 3.2. There are many ways to construct sequences that satisfy the desired relation. One is as follows: Let $l \in \mathbb{R}^{N\times|\mathcal{V}|}$ be any logit sequence with no re-occurring values. Denote by $m_k(l)_i$ the top-k value at position i, and by $a_k(l)_i$ the top-k vocab index at position i, respectively. Now we pick any such sequence with the additional property that $\max_i m_1(l)_i - m_2(l)_i < \delta$ for some small δ. Define the sequence l' by

$$l'_{ia_2(l)_i} = l_{ia_2(l)_i} + \delta,$$

and $l'_{ij} = l_{ij}$ for all remaining indices. Then we have $a_1(l)_i \neq a_1(l')_i$ for all i and hence $M_{\mathrm{SDT}}(l, l', y_{:1}, N) = N$. On the other hand we have $||l - l'||_\infty \leq \delta$. Since $\mathrm{PPL}(y, l, 1)$ is a continuous function in l, we have $|\mathrm{PPL}(y, l, 1) - \mathrm{PPL}(y, l', 1)| < \varepsilon$ for any ε and small enough δ. $\square$

Proof of Proposition 3.3. Let $z = \mathcal{G}(l, y_{:n}, N)$ and $p_i = (\text{softmax } l')_{iz_{i+1}}$. Applying the definitions and elementary operations, we have

$$\sum_{i=n}^{N} -\log p_i = (N-n)\log M_{\mathrm{DPPL}}(l, l', y_{:n}, N).$$

Let $A = \{i \geq n \colon p_i \leq 1/2\}$. Then

$$\begin{aligned}\sum_{i=n}^{N} -\log p_i &= \sum_{i\in A} -\log p_i + \sum_{i\in A^c} -\log p_i \\ &\geq \sum_{i\in A} -\log p_i \geq |A|\log 2.\end{aligned}$$

Here we first used that $\log p_i \leq 0$ and then the observation that indices contained in A satisfy $-\log p_i \geq \log 2$ by the defining property of A. Finally, we argue that $\mathrm{SDT}(z, l', n) \leq |A|$. Indeed, at any position i where $\arg\max_j l'_{ij} \neq z_{i+1}$ it must hold that $p_i \leq 1/2$, since any softmax-value larger than $1/2$ is automatically the maximum value of the distribution, and the softmax operation is monotone. Putting everything together we arrive at the desired inequality. $\square$

B FDT compared to standard model evaluations

Fig. 10 shows a comparison of standard benchmarks (middle) to FDT (right) and PPL (left) when quantizing parts of a model. Often, standard evaluations fail to distinguish between compressed models. Sometimes they even depict better performance—which may be true, when regarding compression as a fine-tuning method and considering the short required token predictions. FDT thoroughly gives discriminative statistics with resprect to the base model, on how much the compressed model equals the original. Note how the error seems to be upper bounded, which suggests that errors may average out throughout the model. Mean zeroshot accuracy denotes the average on the standard NLP-eval harness.

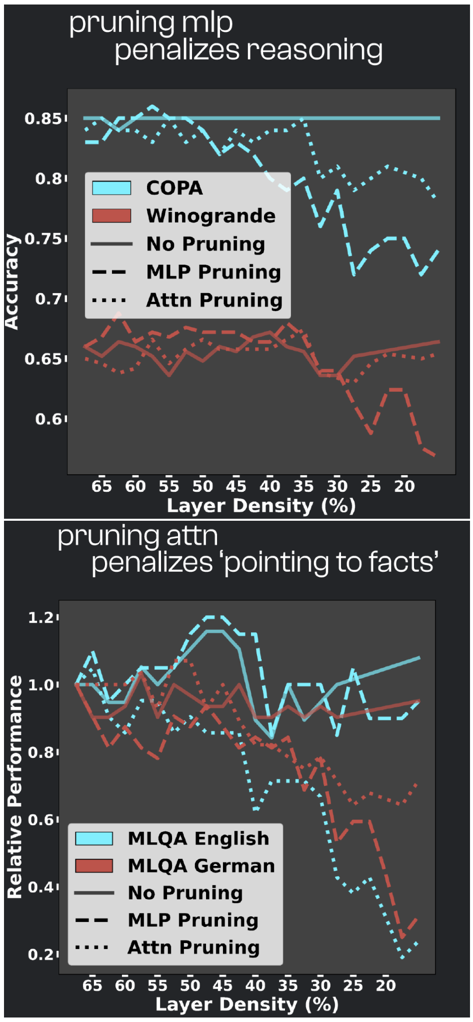


Figure 7: Pruning MLP and Attn only indeed compromises remaining model capabilities.

C True positives can be predicted

Fig. 8 shows several metrics applied to the token-distributions, in order to estimate on whether the compressed and original model predictions are equal. Notably, L1 and L2 errors on the entire distribution seem to somewhat capture the discriminative capabilities of false predictions. The probability scores themselves are only marginally usable. Using top-2 uncertainty, i.e. the difference between the top-2 tokens as a measure, we obtain a reliable prediction of true positives. True negatives however still remain with a significant overlap.

D MLP is for knowledge, Attention for relation

Finally, we observed that when pruning only attention, prompt-extraction capabilities degenerate severely. When only pruning MLP components, on the other hand, it influences mostly world knowledge QA benchmarks, *c.f.* Fig. 7.

E Details on Search Tree, Sec. 4.3

Fig. 9 shows the layers (y-axis) of which components are selected at each round (x-axis). While there seems to be a pattern on when using FDT as a criteria (top), selection by PPL (bottom) looks more random.

Fig. 15 shows the comparison of search tree as described to greedy search on a single evaluation of all components. Until 150 components, FDT proves more stable over the PPL variants as seen in Fig. 15a.

F Details on Quantization Sec. 4.3

Fig. 17 shows detailed component-wise evaluations aggregated in Fig. 6a.

Fig. 16 shows the final configurations as compared in Tab. 1.

Fig. 11 shows the detailed nlp-eval scores of Tab. 1.

Fig. 12 shows greedy search trees over various context lengths.

In total the entire search evaluation required 16 GPU-days with A100s to complete all metrics.

G Details on Sparsification, Sec. 4.2

Fig. 18 shows a different aggregated perspective of Fig. 4b, to point out more direct the occuring variances.

Fig. 19 shows the rank of lowest influence (measured by FDT) of components (x-axis) throughout various sparsity levels (y-axis). I.e. starting with a uniformly pruned model in 5% steps, we measured the rank when adding an additional 2.5% only to a single component. Interestingly, components seem to retain their importance throughout the various levels of sparsity.

Fig. 13 shows the detailed nlp-eval scores of Tab. 1.

Note that, despite being often close in relative sparsity, the total number of parameters pruned for MLP is significantly larger than for Attention matrices (ratio 3:1).

In total one sparsification training required 32 GPU-days with A100s for our experiment, and 29 GPU-days for uniform pruning.

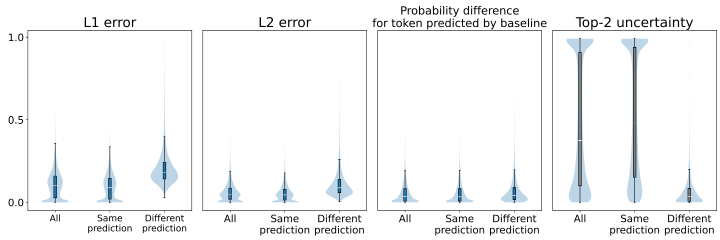


Figure 8: Top-2 uncertainty is discriminative enough to give clear true-positives estimates on compressed models.

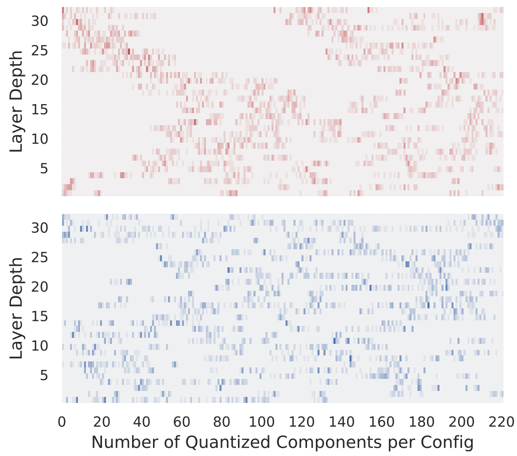


Figure 9: Layers selected in each round of the search tree. Top, when applying FDT, bottom, when applying PPL as a ranking metric.

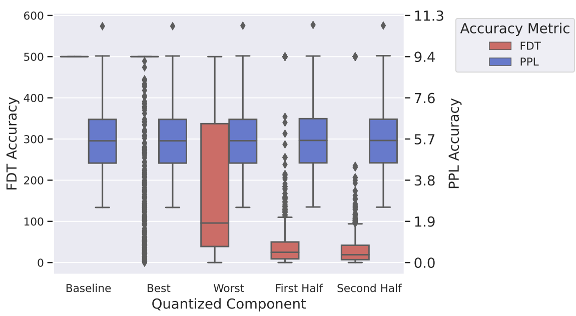
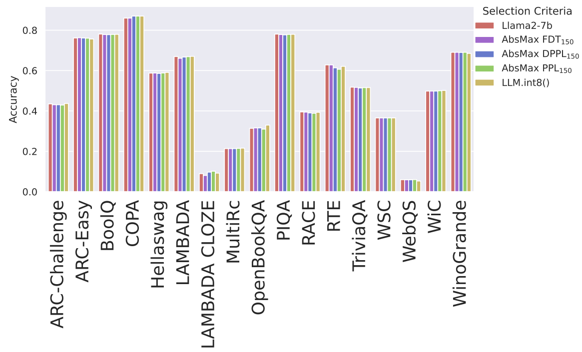
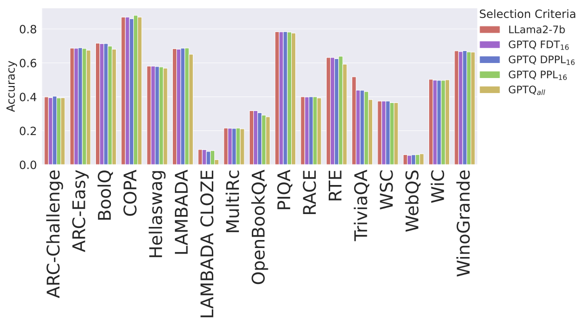


Figure 10: Comparison of the discrimination capabilities of FDT and PPL for different configurations when applying LLM.int8() conversion on Llama2-7B. Best and Worst mark a single component being converted, with most and least mean influence. First and Second half consecutively convert half of the model each. While significant changes can be observed using FDT, all configurations appear indifferent for PPL.

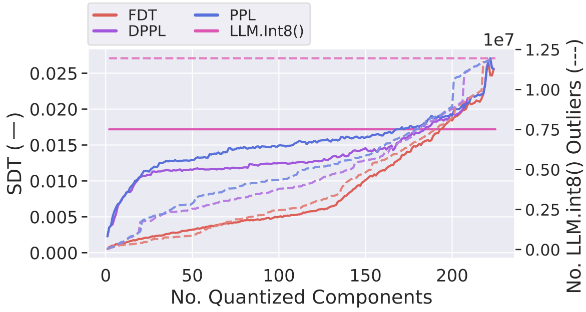


(a) 8-bit Quantization NLP benchmarks

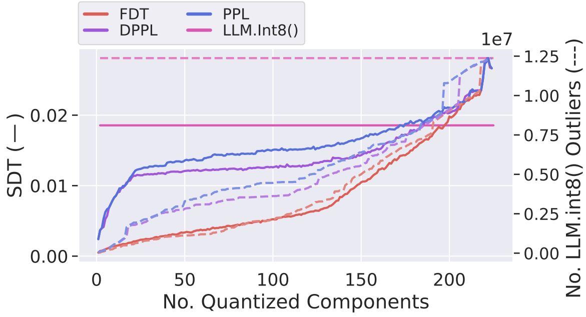


(b) 4-bit Quantization NLP benchmarks

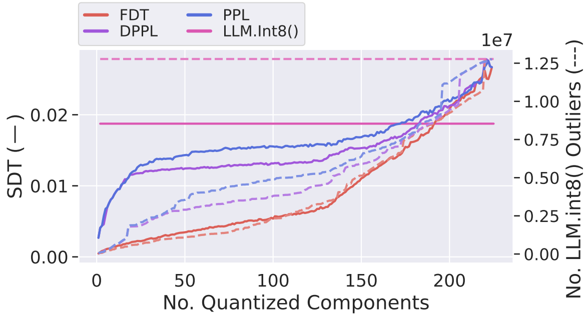
Figure 11: Detailed view on aggregated values of Tab. 1 when selecting Llama2-7B components to quantize by metrics.



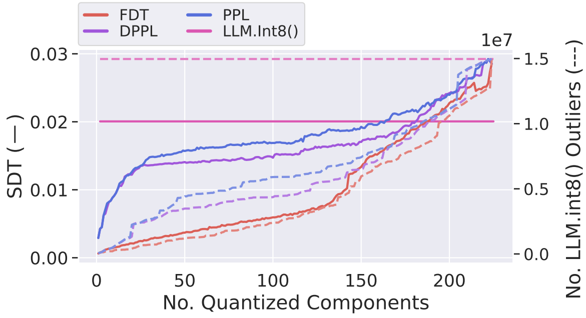
(a) Context size of 10 tokens.



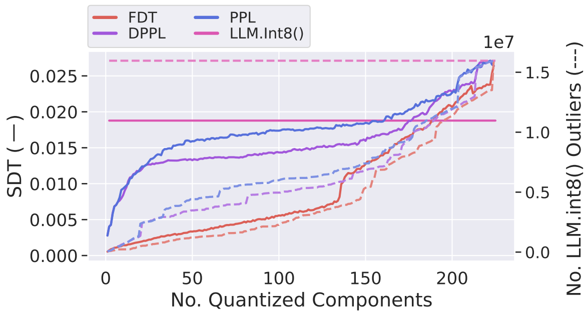
(b) Context size of 25 tokens.



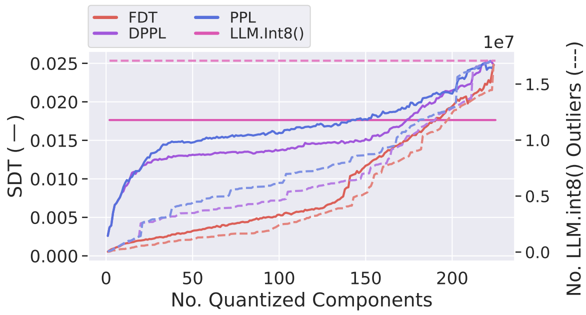
(c) Context size of 50 tokens.



(d) Context size of 200 tokens.



(e) Context size of 300 tokens.



(f) Context size of 400 tokens.

Figure 12: Greedy Search Tree results for different context sizes.

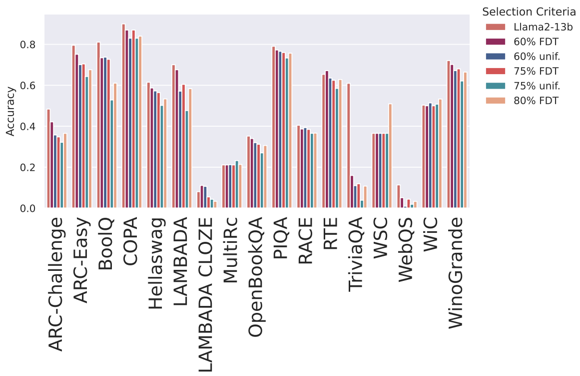
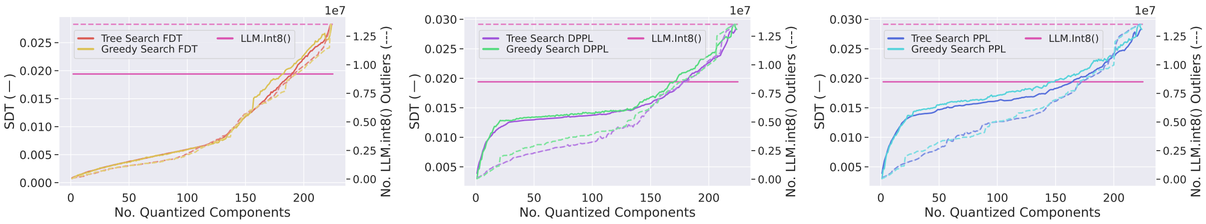


Figure 13: Detailed view on aggregated values of Tab. 1 when selecting Llama2-13B components to sparsify by metrics.

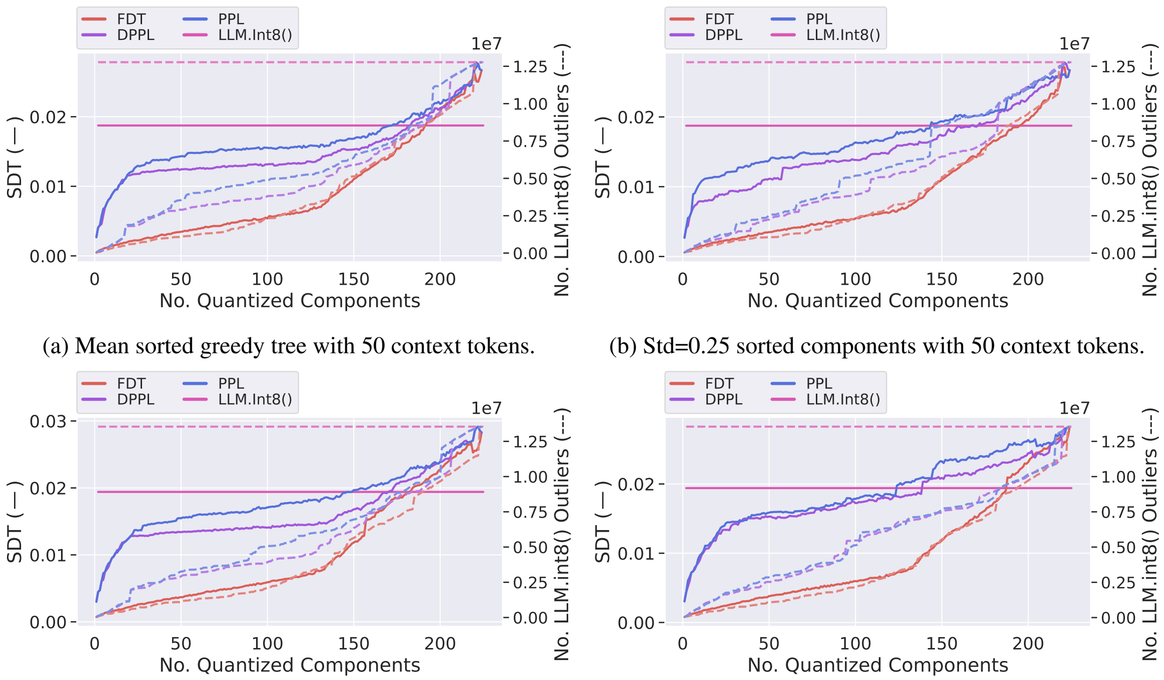


(a) FDT tree vs greedy

(b) DPPL tree vs greedy

(c) PPL tree vs greedy

Figure 14: Comparison of performance when selecting components by the tree-search as described to greedy selection of once evaluated components for all discussed metrics. Clearly, FDT is most stable until 150 components.



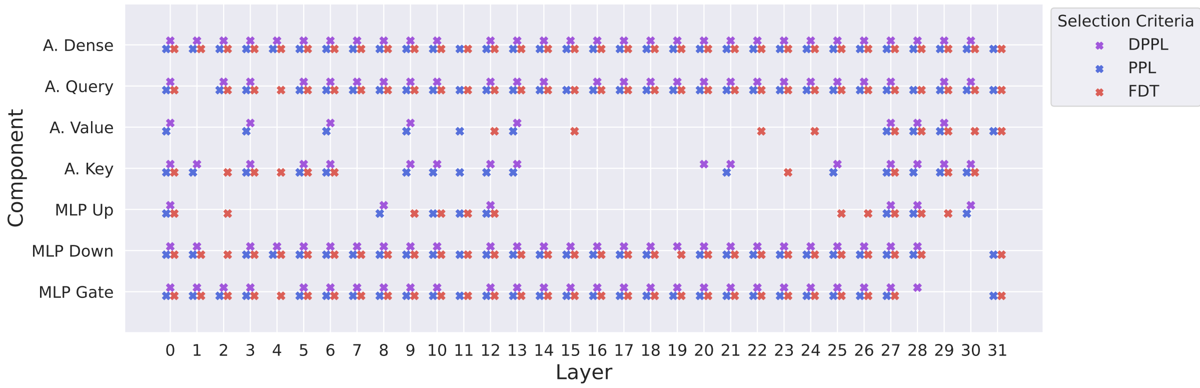
(a) Mean sorted greedy tree with 50 context tokens.

(b) Std=0.25 sorted components with 50 context tokens.

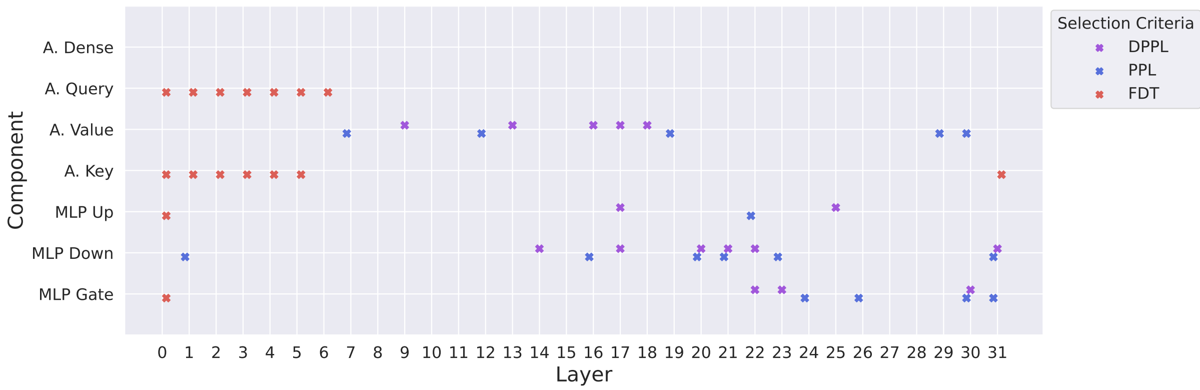
(c) Mean sorted greedy tree with 100 context tokens.

(d) Std=0.25 sorted components with 100 context tokens.

Figure 15: Comparing the ranking of the components based on mean or standard deviation.



(a) The 150 selected components selected by metrics for 8-bit AbsMax conversion.



(b) The 16 selected components selected by metrics for 4-bit GPTQ conversion.

Figure 16: Detailed view of the Llama2-7B components in Tab. 1 selected by metrics for lower precision conversion.

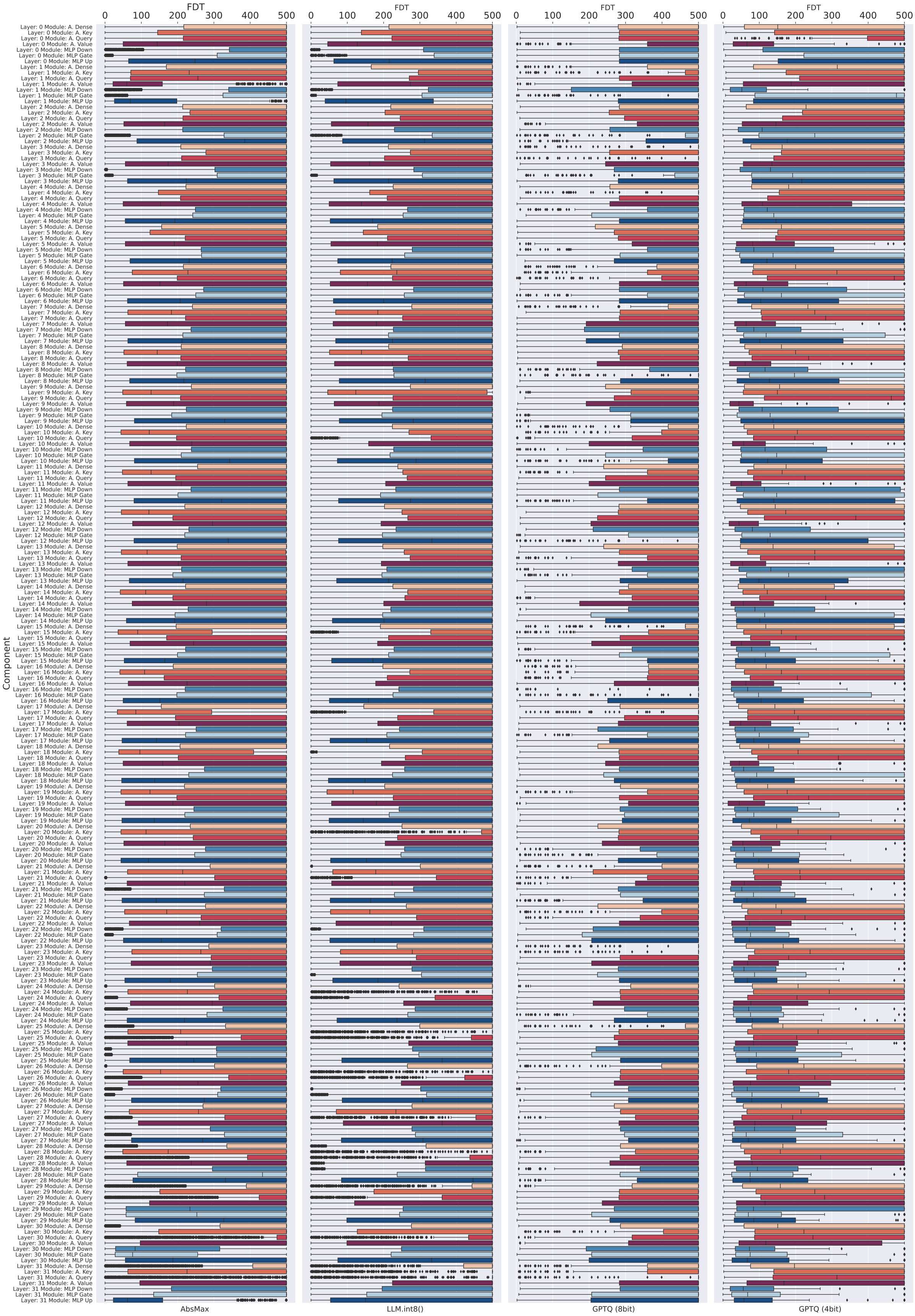


Figure 17: Full view of the influence of individual component-wise quantization measured by FDT.

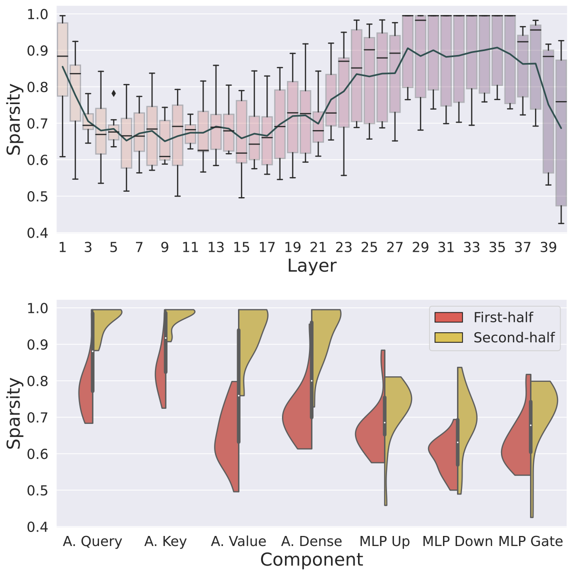


Figure 18: Distribution of 75% average model sparsity. A. denotes Attention. **Top**: Aggregated by layers. The first and last layer have highest variance (MLP most important, c.f. Fig. 4b). Second half reaches sparsities close component removal. **Bottom**: Per component aggregation. In the second half of layers, the importance of attention drops drastically. MLP almost remains, with outliers to larger importance.

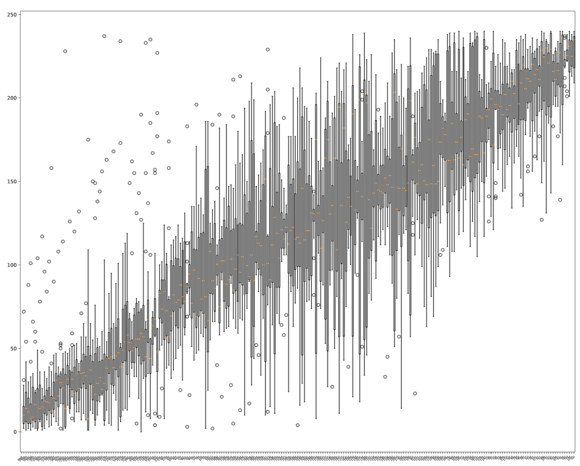


Figure 19: Trends during sparsification. We plot the ranking of the components FDT value through various sparsity levels (y-axis) for all components (x-axis). Interestingly, there is a clear trend of components retaining "their importance".